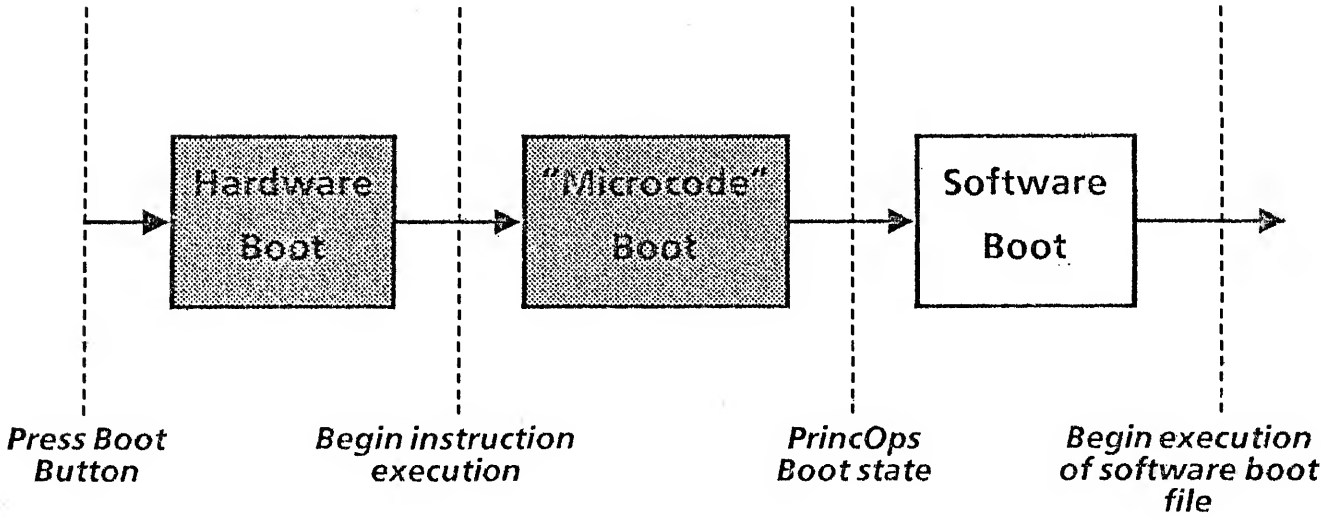
STAR Format

Folder: [OSBUNORTH: Tundra] <DANDELION Development> Booting Slides

Dandelion Booting

- The initialization of the system state (control store, main memory, IOP memory, etc.) to that required before the first software instruction (Mesa opcode) can be executed (the so-called *PrincOps Boot State*)

- Boot Stages



PrincOps Boot State

- Operational CP microcode and IOP control code is loaded into control store and IOP memory
- Pilot Boot File loader (Germ) has been loaded into the appropriate place in main memory
- I/O Page has been mapped to its assigned virtual address
- All other normal real memory (i.e. excluding display bank and VM map) has been mapped
- The Mesa emulator has been prepared to execute the first Mesa byte code of the Germ

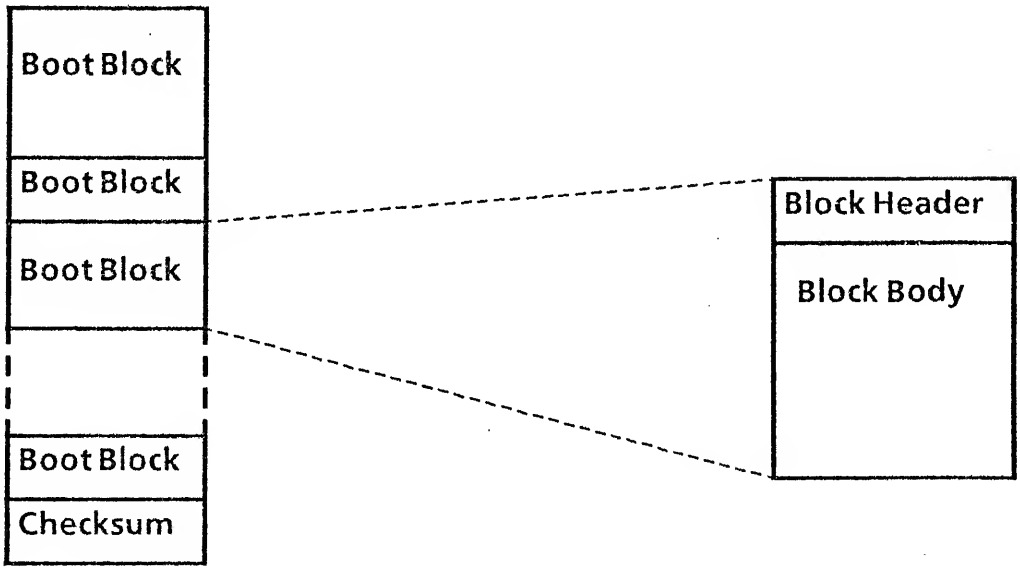
- **Hardware Boot**
  - reset the hardware
  - put the CP in a Wait state
  - cause the IOP to start execution at a specific address in IOP memory
- **"Microcode Boot"**
  - consists of multiple phases
    - saves EProm space
    - allows more flexibility with changes
    - overlays "one time" boot code
    - provides a powerful diagnostic strategy
  - function:
    - delivers Mesa and I/O microcode to control store
    - delivers IOP control program (Domino) to IOP memory
    - delivers Germ to main memory
    - initializes the VM map, and Mesa emulator
  - runs PreBoot diagnostics
  - source of boot files: EProm and Boot Device
  - allows selection of:
    - Boot Device: rigid disk, floppy disk, Ethernet (2 sources)
    - Boot Mode: normal boot or diagnostic boot

- **EProm contents**
  - PreBoot diagnostics
  - IOP boot code
  - Phase 0 CP microcode boot file
  - IOP interrupt links

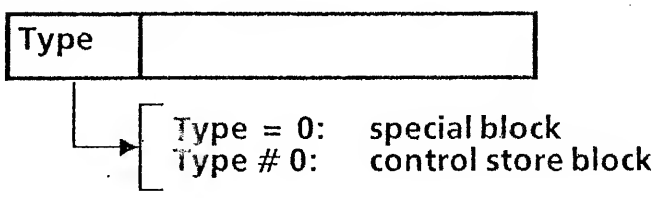
- **Boot concepts**
  - Boot files and boot blocks
  - *Processing* boot files
  - CP execution states
  - CP kernel and CP protected areas

Boot files

- name: *.db
- a series of *boot blocks*, together with a checksum
- boot blocks vary in length
- structure:

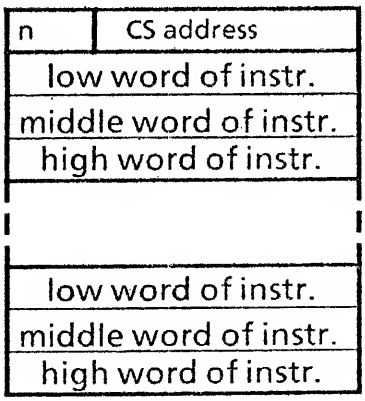


- Block Header indicates type of block:



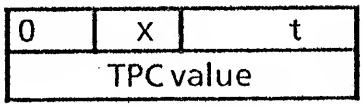
- types of boot blocks:
  - control store
  - special
    - task program counter (TPC)
    - U register
    - IOP memory
    - ignore
    - start IOP address
    - last block
- examples of boot blocks:

Control store block



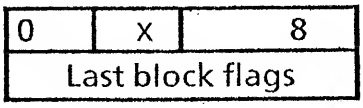
n = [1 .. 15]

TPC block

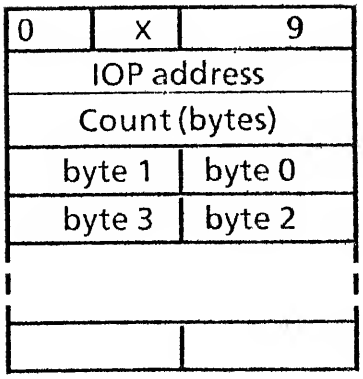


t = [0 .. 7]

Last block



IOP memory block



- **Processing boot files**
  - unpack the boot blocks, i.e. load the data into the appropriate part of the Dandelion hardware
  - special precautions are needed to process the control store and TPC boot blocks
- **CP execution states**
  - Run state: CP is executing in multitask fashion
  - Stopped state: CP is executing in kernel task (7)
  - Wait state: CP is not executing
- **CP kernel**
  - Special CP microprogram that executes while the CP is in the "stopped" state
  - Can communicate with IOP
  - Performs memory refresh

Microcode Boot Phases

- **Phase 0**
  - Run PreBoot diagnostics (IOP)
  - Determine boot device (IOP, CP)
  - Process "Phase0.db" (from EProm) boot file (IOP)
- **Phase 1**
  - Start CP execution of "Phase0" microcode (IOP)
    - ⇒ Load "Initial.db" boot file (from Boot device) into main memory (CP)
  - Process "Initial.db" (from main memory) file (IOP)
- **Phase 2**
  - Start CP execution of "Initial" microcode (IOP)
    - ⇒ Load "Mesa.db" boot file (from Boot device) into main memory. This contains Mesa and I/O microcode, and Domino code (CP)
    - ⇒ Load the Germ into main memory (CP)
    - ⇒ Initialize the VM map (CP)
    - ⇒ Initialize the Mesa emulator (CP)
  - Process "Mesa.db" (from main memory) file (IOP)

- **Transfer to Software booting**

  - Start CP execution of Mesa, I/O microcode (IOP)
    ⇒ Germ software starts executing (CP)
  - Transfer to start of Domino code (IOP)
    ⇒ Domino software starts executing (IOP)

- **Boot files location (rigid disk booting)**

  - Phase0 boot file: located in EProm
  - Initial boot file: located at fixed location on disk, stored in consecutive sectors [cyl 0, hd 1, sec 0 on]
  - Physical Volume Root Page (PVRP):
    points to rest of boot files
    located at fixed location on disk [cyl 0, hd 0, sec 0]
    contains pointers to:
    → Mesa microcode boot file
    → Germ boot file
    → Diagnostic boot file
    → Software boot file (e.g. Othello)

- **Protected areas in CP**

  - Control store and TPC images

    Low 128 locations in control store contain CP "vital" functions of IOP task code, memory refresh code, trap-catcher code, loop code

    Image of protected control store area is kept in IOP memory

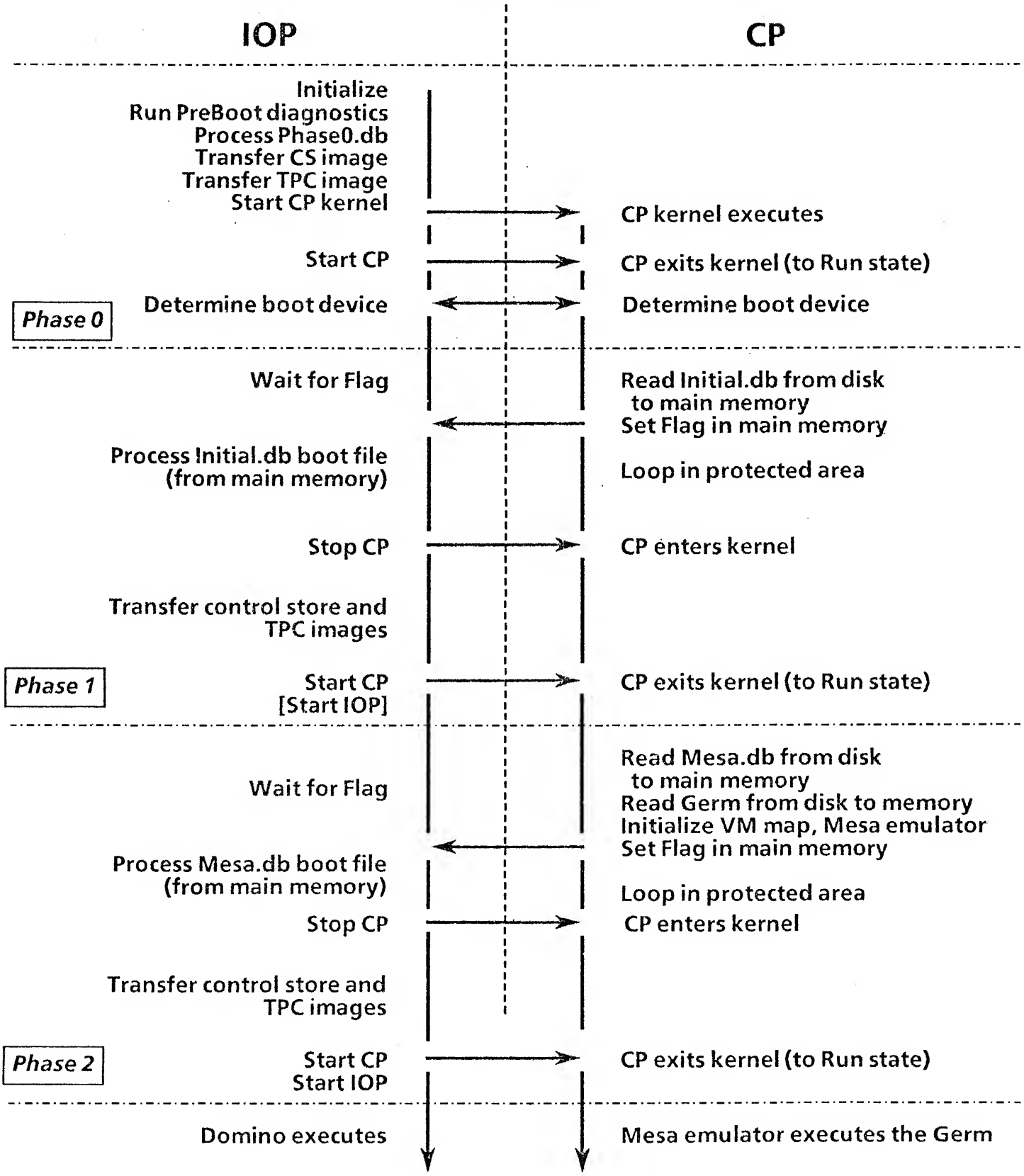
    Image of TPC values is kept in IOP memory

    Images are transferred to CP at end of boot phase

    Writing control store and TPC values requires CP to be stopped AND requires careful CP-IOP interaction to avoid losing memory refresh

  - Kernel microcode is never overlaid (32 locations)

Detailed description of rigid disk booting



Other topics

- **Floppy booting**
  - Similar structure to rigid disk booting except that the boot device is directly accessible from the IOP
  - The IOP processes boot files directly from the floppy disk
- **Ethernet booting (Etherbooting)**
  - Similar structure to rigid disk booting
  - Boot files are fetched from boot server over the Ethernet
- **Diagnostic booting**
  - Allows the repeated execution of a collection of special boot diagnostics
  - Diagnostics have total control of machine, but use the boot code to fetch and process boot files
  - Implemented by repeating Phase 2 multiple times

Appendix

- **Alternate boot codes**

AltBoot code	Type of boot
0	rigid* (diagnostic)
1	rigid*
2	floppy
3	ethernet
4	ethernet (diagnostic)
5	floppy (diagnostic)
6	alternate ethernet
7	Trident 1 (diagnostic)
8	Trident 2 (diagnostic)
9	Trident 3 (diagnostic)
10	floppy cleaning

* AltBoot 0 and 1 apply to the Shugart 1004, SA4008, Quantum, or Trident 0 disk, whichever is installed in the system